

APPLICATION  
FOR  
UNITED STATES LETTERS PATENT

10559-618001-04999  
2002-01-25

TITLE: DATA TRANSFER MECHANISM

APPLICANT: GILBERT WOLRICH, MARK B. ROSENBLUTH, DEBRA  
BERNSTEIN AND MATTHEW J. ADILETTA

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EL870691429US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

January 25, 2002

Date of Deposit

Signature

Gabriel Lewis

Typed or Printed Name of Person Signing Certificate

## DATA TRANSFER MECHANISM

### BACKGROUND

Typical computer processing systems have buses that enable  
5 various components to communicate with each other. Bus  
communication between these components allow transfer of data  
commonly through a data path. Generally, the datapath  
interconnects a processing agent, e.g., a central processing unit  
(CPU) or processor, with other components such as hard disk  
10 drives, device adapters, and the like.

### BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a processing system.

FIG. 2 is a detailed block diagram of the processing system  
of FIG. 1.

FIG. 3 is a flow chart of a read process in the processing  
system of FIG. 1.

FIG. 4 is a flow chart of a write process in the processing  
system of FIG. 1.

20 FIG. 5 is a flow chart of a push operation of the processing  
system of FIG. 1.

FIG. 6 is a flow chart of a pull operation of the processing  
system of FIG. 1.

## DESCRIPTION

### Architecture:

Referring to FIG. 1, a computer processing system 10 includes a parallel, hardware-based multithreaded network processor 12. The hardware-based multithreaded processor 12 is coupled to a memory system or memory resource 14. Memory system 14 includes dynamic random access memory (DRAM) 14a and static random access memory 14b (SRAM). The processing system 10 is especially useful for tasks that can be broken into parallel subtasks or functions. Specifically, the hardware-based multithreaded processor 12 is useful for tasks that are bandwidth oriented rather than latency oriented. The hardware-based multithreaded processor 12 has multiple microengines or programming engines 16 each with multiple hardware controlled threads that are simultaneously active and independently work on a specific task.

The programming engines 16 each maintain program counters in hardware and states associated with the program counters.

Effectively, corresponding sets of context or threads can be simultaneously active on each of the programming engines 16 while only one is actually operating at any one time.

In this example, eight programming engines 16 are

illustrated in FIG. 1. Each programming engine 16 has capabilities for processing eight hardware threads or contexts. The eight programming engines 16 operate with shared resources including memory resource 14 and bus interfaces. The hardware-based multithreaded processor 12 includes a dynamic random access memory (DRAM) controller 18a and a static random access memory (SRAM) controller 18b. The DRAM memory 14a and DRAM controller 18a are typically used for processing large volumes of data, e.g., processing of network payloads from network packets. The SRAM memory 14b and SRAM controller 18b are used in a networking implementation for low latency, fast access tasks, e.g., accessing look-up tables, memory for the core processor 20, and the like.

Push buses 26a-26b and pull buses 28a-28b are used to transfer data between the programming engines 16 and the DRAM memory 14a and the SRAM memory 14b. In particular, the push buses 26a-26b are unidirectional buses that move the data from the memory resources 14 to the programming engines 16 whereas the pull buses 28a-28b move data from the programming engines 16 to the memory resources 14.

The eight programming engines 16 access either the DRAM memory 14a or SRAM memory 14b based on characteristics of the data. Thus, low latency, low bandwidth data are stored in and

5 fetched from SRAM memory 14b, whereas higher bandwidth data for which latency is not as important, are stored in and fetched from DRAM 14a. The programming engines 16 can execute memory reference instructions to either the DRAM controller 18a or SRAM controller 18b.

The hardware-based multithreaded processor 12 also includes a processor core 20 for loading microcode control for other resources of the hardware-based multithreaded processor 12. In this example, the processor core 20 is an XScale™ based architecture.

10  
15  
20  
25  
30  
35  
40  
45  
50  
55  
60  
65  
70  
75  
80  
85  
90  
95  
100  
105  
110  
115  
120  
125  
130  
135  
140  
145  
150  
155  
160  
165  
170  
175  
180  
185  
190  
195  
200  
205  
210  
215  
220  
225  
230  
235  
240  
245  
250  
255  
260  
265  
270  
275  
280  
285  
290  
295  
300  
305  
310  
315  
320  
325  
330  
335  
340  
345  
350  
355  
360  
365  
370  
375  
380  
385  
390  
395  
400  
405  
410  
415  
420  
425  
430  
435  
440  
445  
450  
455  
460  
465  
470  
475  
480  
485  
490  
495  
500  
505  
510  
515  
520  
525  
530  
535  
540  
545  
550  
555  
560  
565  
570  
575  
580  
585  
590  
595  
600  
605  
610  
615  
620  
625  
630  
635  
640  
645  
650  
655  
660  
665  
670  
675  
680  
685  
690  
695  
700  
705  
710  
715  
720  
725  
730  
735  
740  
745  
750  
755  
760  
765  
770  
775  
780  
785  
790  
795  
800  
805  
810  
815  
820  
825  
830  
835  
840  
845  
850  
855  
860  
865  
870  
875  
880  
885  
890  
895  
900  
905  
910  
915  
920  
925  
930  
935  
940  
945  
950  
955  
960  
965  
970  
975  
980  
985  
990  
995  
1000  
1005  
1010  
1015  
1020  
1025  
1030  
1035  
1040  
1045  
1050  
1055  
1060  
1065  
1070  
1075  
1080  
1085  
1090  
1095  
1100  
1105  
1110  
1115  
1120  
1125  
1130  
1135  
1140  
1145  
1150  
1155  
1160  
1165  
1170  
1175  
1180  
1185  
1190  
1195  
1200  
1205  
1210  
1215  
1220  
1225  
1230  
1235  
1240  
1245  
1250  
1255  
1260  
1265  
1270  
1275  
1280  
1285  
1290  
1295  
1300  
1305  
1310  
1315  
1320  
1325  
1330  
1335  
1340  
1345  
1350  
1355  
1360  
1365  
1370  
1375  
1380  
1385  
1390  
1395  
1400  
1405  
1410  
1415  
1420  
1425  
1430  
1435  
1440  
1445  
1450  
1455  
1460  
1465  
1470  
1475  
1480  
1485  
1490  
1495  
1500  
1505  
1510  
1515  
1520  
1525  
1530  
1535  
1540  
1545  
1550  
1555  
1560  
1565  
1570  
1575  
1580  
1585  
1590  
1595  
1600  
1605  
1610  
1615  
1620  
1625  
1630  
1635  
1640  
1645  
1650  
1655  
1660  
1665  
1670  
1675  
1680  
1685  
1690  
1695  
1700  
1705  
1710  
1715  
1720  
1725  
1730  
1735  
1740  
1745  
1750  
1755  
1760  
1765  
1770  
1775  
1780  
1785  
1790  
1795  
1800  
1805  
1810  
1815  
1820  
1825  
1830  
1835  
1840  
1845  
1850  
1855  
1860  
1865  
1870  
1875  
1880  
1885  
1890  
1895  
1900  
1905  
1910  
1915  
1920  
1925  
1930  
1935  
1940  
1945  
1950  
1955  
1960  
1965  
1970  
1975  
1980  
1985  
1990  
1995  
2000  
2005  
2010  
2015  
2020  
2025  
2030  
2035  
2040  
2045  
2050  
2055  
2060  
2065  
2070  
2075  
2080  
2085  
2090  
2095  
2100  
2105  
2110  
2115  
2120  
2125  
2130  
2135  
2140  
2145  
2150  
2155  
2160  
2165  
2170  
2175  
2180  
2185  
2190  
2195  
2200  
2205  
2210  
2215  
2220  
2225  
2230  
2235  
2240  
2245  
2250  
2255  
2260  
2265  
2270  
2275  
2280  
2285  
2290  
2295  
2300  
2305  
2310  
2315  
2320  
2325  
2330  
2335  
2340  
2345  
2350  
2355  
2360  
2365  
2370  
2375  
2380  
2385  
2390  
2395  
2400  
2405  
2410  
2415  
2420  
2425  
2430  
2435  
2440  
2445  
2450  
2455  
2460  
2465  
2470  
2475  
2480  
2485  
2490  
2495  
2500  
2505  
2510  
2515  
2520  
2525  
2530  
2535  
2540  
2545  
2550  
2555  
2560  
2565  
2570  
2575  
2580  
2585  
2590  
2595  
2600  
2605  
2610  
2615  
2620  
2625  
2630  
2635  
2640  
2645  
2650  
2655  
2660  
2665  
2670  
2675  
2680  
2685  
2690  
2695  
2700  
2705  
2710  
2715  
2720  
2725  
2730  
2735  
2740  
2745  
2750  
2755  
2760  
2765  
2770  
2775  
2780  
2785  
2790  
2795  
2800  
2805  
2810  
2815  
2820  
2825  
2830  
2835  
2840  
2845  
2850  
2855  
2860  
2865  
2870  
2875  
2880  
2885  
2890  
2895  
2900  
2905  
2910  
2915  
2920  
2925  
2930  
2935  
2940  
2945  
2950  
2955  
2960  
2965  
2970  
2975  
2980  
2985  
2990  
2995  
3000  
3005  
3010  
3015  
3020  
3025  
3030  
3035  
3040  
3045  
3050  
3055  
3060  
3065  
3070  
3075  
3080  
3085  
3090  
3095  
3100  
3105  
3110  
3115  
3120  
3125  
3130  
3135  
3140  
3145  
3150  
3155  
3160  
3165  
3170  
3175  
3180  
3185  
3190  
3195  
3200  
3205  
3210  
3215  
3220  
3225  
3230  
3235  
3240  
3245  
3250  
3255  
3260  
3265  
3270  
3275  
3280  
3285  
3290  
3295  
3300  
3305  
3310  
3315  
3320  
3325  
3330  
3335  
3340  
3345  
3350  
3355  
3360  
3365  
3370  
3375  
3380  
3385  
3390  
3395  
3400  
3405  
3410  
3415  
3420  
3425  
3430  
3435  
3440  
3445  
3450  
3455  
3460  
3465  
3470  
3475  
3480  
3485  
3490  
3495  
3500  
3505  
3510  
3515  
3520  
3525  
3530  
3535  
3540  
3545  
3550  
3555  
3560  
3565  
3570  
3575  
3580  
3585  
3590  
3595  
3600  
3605  
3610  
3615  
3620  
3625  
3630  
3635  
3640  
3645  
3650  
3655  
3660  
3665  
3670  
3675  
3680  
3685  
3690  
3695  
3700  
3705  
3710  
3715  
3720  
3725  
3730  
3735  
3740  
3745  
3750  
3755  
3760  
3765  
3770  
3775  
3780  
3785  
3790  
3795  
3800  
3805  
3810  
3815  
3820  
3825  
3830  
3835  
3840  
3845  
3850  
3855  
3860  
3865  
3870  
3875  
3880  
3885  
3890  
3895  
3900  
3905  
3910  
3915  
3920  
3925  
3930  
3935  
3940  
3945  
3950  
3955  
3960  
3965  
3970  
3975  
3980  
3985  
3990  
3995  
4000  
4005  
4010  
4015  
4020  
4025  
4030  
4035  
4040  
4045  
4050  
4055  
4060  
4065  
4070  
4075  
4080  
4085  
4090  
4095  
4100  
4105  
4110  
4115  
4120  
4125  
4130  
4135  
4140  
4145  
4150  
4155  
4160  
4165  
4170  
4175  
4180  
4185  
4190  
4195  
4200  
4205  
4210  
4215  
4220  
4225  
4230  
4235  
4240  
4245  
4250  
4255  
4260  
4265  
4270  
4275  
4280  
4285  
4290  
4295  
4300  
4305  
4310  
4315  
4320  
4325  
4330  
4335  
4340  
4345  
4350  
4355  
4360  
4365  
4370  
4375  
4380  
4385  
4390  
4395  
4400  
4405  
4410  
4415  
4420  
4425  
4430  
4435  
4440  
4445  
4450  
4455  
4460  
4465  
4470  
4475  
4480  
4485  
4490  
4495  
4500  
4505  
4510  
4515  
4520  
4525  
4530  
4535  
4540  
4545  
4550  
4555  
4560  
4565  
4570  
4575  
4580  
4585  
4590  
4595  
4600  
4605  
4610  
4615  
4620  
4625  
4630  
4635  
4640  
4645  
4650  
4655  
4660  
4665  
4670  
4675  
4680  
4685  
4690  
4695  
4700  
4705  
4710  
4715  
4720  
4725  
4730  
4735  
4740  
4745  
4750  
4755  
4760  
4765  
4770  
4775  
4780  
4785  
4790  
4795  
4800  
4805  
4810  
4815  
4820  
4825  
4830  
4835  
4840  
4845  
4850  
4855  
4860  
4865  
4870  
4875  
4880  
4885  
4890  
4895  
4900  
4905  
4910  
4915  
4920  
4925  
4930  
4935  
4940  
4945  
4950  
4955  
4960  
4965  
4970  
4975  
4980  
4985  
4990  
4995  
5000  
5005  
5010  
5015  
5020  
5025  
5030  
5035  
5040  
5045  
5050  
5055  
5060  
5065  
5070  
5075  
5080  
5085  
5090  
5095  
5100  
5105  
5110  
5115  
5120  
5125  
5130  
5135  
5140  
5145  
5150  
5155  
5160  
5165  
5170  
5175  
5180  
5185  
5190  
5195  
5200  
5205  
5210  
5215  
5220  
5225  
5230  
5235  
5240  
5245  
5250  
5255  
5260  
5265  
5270  
5275  
5280  
5285  
5290  
5295  
5300  
5305  
5310  
5315  
5320  
5325  
5330  
5335  
5340  
5345  
5350  
5355  
5360  
5365  
5370  
5375  
5380  
5385  
5390  
5395  
5400  
5405  
5410  
5415  
5420  
5425  
5430  
5435  
5440  
5445  
5450  
5455  
5460  
5465  
5470  
5475  
5480  
5485  
5490  
5495  
5500  
5505  
5510  
5515  
5520  
5525  
5530  
5535  
5540  
5545  
5550  
5555  
5560  
5565  
5570  
5575  
5580  
5585  
5590  
5595  
5600  
5605  
5610  
5615  
5620  
5625  
5630  
5635  
5640  
5645  
5650  
5655  
5660  
5665  
5670  
5675  
5680  
5685  
5690  
5695  
5700  
5705  
5710  
5715  
5720  
5725  
5730  
5735  
5740  
5745  
5750  
5755  
5760  
5765  
5770  
5775  
5780  
5785  
5790  
5795  
5800  
5805  
5810  
5815  
5820  
5825  
5830  
5835  
5840  
5845  
5850  
5855  
5860  
5865  
5870  
5875  
5880  
5885  
5890  
5895  
5900  
5905  
5910  
5915  
5920  
5925  
5930  
5935  
5940  
5945  
5950  
5955  
5960  
5965  
5970  
5975  
5980  
5985  
5990  
5995  
6000  
6005  
6010  
6015  
6020  
6025  
6030  
6035  
6040  
6045  
6050  
6055  
6060  
6065  
6070  
6075  
6080  
6085  
6090  
6095  
6100  
6105  
6110  
6115  
6120  
6125  
6130  
6135  
6140  
6145  
6150  
6155  
6160  
6165  
6170  
6175  
6180  
6185  
6190  
6195  
6200  
6205  
6210  
6215  
6220  
6225  
6230  
6235  
6240  
6245  
6250  
6255  
6260  
6265  
6270  
6275  
6280  
6285  
6290  
6295  
6300  
6305  
6310  
6315  
6320  
6325  
6330  
6335  
6340  
6345  
6350  
6355  
6360  
6365  
6370  
6375  
6380  
6385  
6390  
6395  
6400  
6405  
6410  
6415  
6420  
6425  
6430  
6435  
6440  
6445  
6450  
6455  
6460  
6465  
6470  
6475  
6480  
6485  
6490  
6495  
6500  
6505  
6510  
6515  
6520  
6525  
6530  
6535  
6540  
6545  
6550  
6555  
6560  
6565  
6570  
6575  
6580  
6585  
6590  
6595  
6600  
6605  
6610  
6615  
6620  
6625  
6630  
6635  
6640  
6645  
6650  
6655  
6660  
6665  
6670  
6675  
6680  
6685  
6690  
6695  
6700  
6705  
6710  
6715  
6720  
6725  
6730  
6735  
6740  
6745  
6750  
6755  
6760  
6765  
6770  
6775  
6780  
6785  
6790  
6795  
6800  
6805  
6810  
6815  
6820  
6825  
6830  
6835  
6840  
6845  
6850  
6855  
6860  
6865  
6870  
6875  
6880  
6885  
6890  
6895  
6900  
6905  
6910  
6915  
6920  
6925  
6930  
6935  
6940  
6945  
6950  
6955  
6960  
6965  
6970  
6975  
6980  
6985  
6990  
6995  
7000  
7005  
7010  
7015  
7020  
7025  
7030  
7035  
7040  
7045  
7050  
7055  
7060  
7065  
7070  
7075  
7080  
7085  
7090  
7095  
7100  
7105  
7110  
7115  
7120  
7125  
7130  
7135  
7140  
7145  
7150  
7155  
7160  
7165  
7170  
7175  
7180  
7185  
7190  
7195  
7200  
7205  
7210  
7215  
7220  
7225  
7230  
7235  
7240  
7245  
7250  
7255  
7260  
7265  
7270  
7275  
7280  
7285  
7290  
7295  
7300  
7305  
7310  
7315  
7320  
7325  
7330  
7335  
7340  
7345  
7350  
7355  
7360  
7365  
7370  
7375  
7380  
7385  
7390  
7395  
7400  
7405  
7410  
7415  
7420  
7425  
7430  
7435  
7440  
7445  
7450  
7455  
7460  
7465  
7470  
7475  
7480  
7485  
7490  
7495  
7500  
7505  
7510  
7515  
7520  
7525  
7530  
7535  
7540  
7545  
7550  
7555  
7560  
7565  
7570  
7575  
7580  
7585  
7590  
7595  
7600  
7605  
7610  
7615  
7620  
7625  
7630  
7635  
7640  
7645  
7650  
7655  
7660  
7665  
7670  
7675  
7680  
7685  
7690  
7695  
7700  
7705  
7710  
7715  
7720  
7725  
7730  
7735  
7740  
7745  
7750  
7755  
7760  
7765  
7770  
7775  
7780  
7785  
7790  
7795  
7800  
7805  
7810  
7815  
7820  
7825  
7830  
7835  
7840  
7845  
7850  
7855  
7860  
7865  
7870  
7875  
7880  
7885  
7890  
7895  
7900  
7905  
7910  
7915  
7920  
7925  
7930  
7935  
7940  
7945  
7950  
7955  
7960  
7965  
7970  
7975  
7980  
7985  
7990  
7995  
8000  
8005  
8010  
8015  
8020  
8025  
8030  
8035  
8040  
8045  
8050  
8055  
8060  
8065  
8070  
8075  
8080  
8085  
8090  
8095  
8100  
8105  
8110  
8115  
8120  
8125  
8130  
8135  
8140  
8145  
8150  
8155  
8160  
8165  
8170  
8175  
8180  
8185  
8190  
8195  
8200  
8205  
8210  
8215  
8220  
8225  
8230  
8235  
8240  
8245  
8250  
8255  
8260  
8265  
8270  
8275  
8280  
8285  
8290  
8295  
8300  
8305  
8310  
8315  
8320  
8325  
8330  
8335  
8340  
8345  
8350  
8355  
8360  
8365  
8370  
8375  
8380  
8385  
8390  
8395  
8400  
8405  
8410  
8415  
8420  
8425  
8430  
8435  
8440  
8445  
8450  
8455  
8460  
8465  
8470  
8475  
8480  
8485  
8490  
8495  
8500  
8505  
8510  
8515  
8520  
8525  
8530  
8535  
8540  
8545  
8550  
8555  
8560  
8565  
8570  
8575  
8580  
8585  
8590  
8595  
8600  
8605  
8610  
8615  
8620  
8625  
8630  
8635  
8640  
8645  
8650  
8655  
8660  
8665  
8670  
8675  
8680  
8685  
8690  
8695  
8700  
8705  
8710  
8715  
8720  
8725  
8730  
8735  
8740  
8745  
8750  
8755  
8760  
8765  
8770  
8775  
8780  
8785  
8790  
8795  
8800  
8805  
8810  
8815  
8820  
8825  
8830  
8835  
8840  
8845  
8850  
8855  
8860  
8865  
8870  
8875  
8880  
8885  
8890  
8895  
8900  
8905  
8910  
8915  
8920  
8925  
8930  
8935  
8940  
8945  
8950  
8955  
8960  
8965  
8970  
8975  
8980  
8985  
8990  
8995  
9000  
9005  
9010  
9015  
9020  
9025  
9030  
9035  
9040  
9045  
9050  
9055  
9060  
9065  
9070  
9075  
9080  
9085  
9090  
9095  
9100  
9105  
9110  
9115  
9120  
9125  
9130  
9135  
9140  
9145  
9150  
9155  
9160  
9165  
9170  
9175  
9180  
9185  
9190  
9195  
9200  
9205  
9210  
9215  
9220  
9225  
9230  
9235  
9240  
9245  
9250  
9255  
9260  
9265  
9270  
9275  
9280  
9285  
9290  
9295  
9300  
9305  
9310  
9315  
9320  
9325  
9330  
9335  
9340  
9345  
9350  
9355  
9360  
9365  
9370  
9375  
9380  
9385  
9390  
9395  
9400  
9405  
9410  
9415  
9420  
9425  
9430  
9435  
9440  
9445  
9450  
9455  
9460  
9465  
9470  
9475  
9480  
9485  
9490  
9495  
9500  
9505  
9510  
9515  
9520  
9525  
9530  
9535  
9540  
9545  
9550  
9555  
9560  
9565  
9570  
9575  
9580  
9585  
9590  
9595  
9600  
9605  
9610  
9615  
9620  
9625  
9630  
9635  
9640  
9645  
9650  
9655  
9660  
9665  
9670  
9675  
9680  
9685  
9690  
9695  
9700  
9705  
9710  
9715  
9720  
9725  
9730  
9735  
9740  
9745  
9750  
9755  
9760  
9765  
9770  
9775  
9780  
9785  
9790  
9795  
9800  
9805  
9810  
9815  
9820  
9825  
9830  
9835  
9840  
9845  
9850  
9855  
9860  
9865  
9870  
9875  
9880  
9885  
9890  
9895  
9900  
9905  
9910  
9915  
9920  
9925  
9930  
9935  
9940  
9945  
9950  
9955  
9960  
9965  
9970  
9975  
9980  
9985  
9990  
9995  
10000  
10005  
10010  
10015  
10020  
10025  
10030  
10035  
10040  
10045  
10050  
10055  
10060  
10065  
10070  
10075  
10080  
10085  
10090  
10095  
10100  
10105  
10110  
10115  
10120  
10125  
10130  
10135  
10140  
10145  
10150  
10155  
10160  
10165  
10170  
10175  
10180  
10185  
10190  
10195  
10200  
10205  
10210  
10215  
10220  
10225  
10230  
10235  
10240  
1

SRAM or DRAM memory accesses. As an example, an SRAM access requested by a context (e.g., Thread\_0), from one of the programming engines 16 will cause the SRAM controller 18b to initiate an access to the SRAM memory 14b. The SRAM controller 18b accesses the SRAM memory 14b, fetches the data from the SRAM memory 14b, and returns data to a requesting programming engine 16.

During an SRAM access, if one of the programming engines 16 had only a single thread that could operate, that programming engine would be dormant until data was returned from the SRAM memory 14b.

By employing hardware context swapping within each of the programming engines 16, the hardware context swapping enables other contexts with unique program counters to execute in that same programming engine. Thus, another thread e.g., Thread\_1 can function while the first thread, Thread\_0, is awaiting the read data to return. During execution, Thread\_1 may access the DRAM memory 14a. While Thread\_1 operates on the DRAM unit, and Thread\_0 is operating on the SRAM unit, a new thread, e.g., Thread\_2 can now operate in the programming engine 16. Thread\_2 can operate for a certain amount of time until it needs to access memory or perform some other long latency operation, such as making an access to a bus interface. Therefore, simultaneously,

the processor 12 can have a bus operation, SRAM operation and DRAM operation all being completed or operated upon by one of the programming engines 16 and have one more thread available to process more work.

5       The hardware context swapping also synchronizes completion of tasks. For example, two threads could hit the shared memory resource, e.g., the SRAM memory 14b. Each one of the separate functional units, e.g., the SRAM controller 18b, and the DRAM controller 18a, when they complete a requested task from one of the programming engine thread or contexts reports back a flag signaling completion of an operation. When the programming engine 16 receives the flag, the programming engine 16 can determine which thread to turn on.

10559-618001-P12857  
16  
One example of an application for the hardware-based multithreaded processor 12 is as a network processor. As a network processor, the hardware-based multithreaded processor 12 interfaces to network devices such as a Media Access Controller (MAC) device, e.g., a 10/100BaseT Octal MAC 13a or a Gigabit Ethernet device (not shown). In general, as a network processor,  
20 the hardware-based multithreaded processor 12 can interface to any type of communication device or interface that receives or sends large amount of data. The computer processing system 10 functioning in a networking application could receive network

packets and process those packets in a parallel manner.

Programming Engine Contexts:

As described above, each of the programming engines 16  
5 supports multi-threaded execution of eight contexts. This allows  
one thread to start executing just after another thread issues a  
memory reference and must wait until that reference completes  
before doing more work. Multi-threaded execution is critical to  
maintaining efficient hardware execution of the programming  
engines 16 because memory latency is significant. Multi-threaded  
execution allows the programming engines 16 to hide memory  
latency by performing useful independent work across several  
threads.

Each of the eight contexts of the programming engines 16, to  
allow for efficient context swapping, has its own register set,  
program counter, and context specific local registers. Having a  
copy per context eliminates the need to move context specific  
information to and from shared memory and programming engine  
registers for each context swap. Fast context swapping allows a  
20 context to perform computations while other contexts wait for  
input-output (I/O), typically, external memory accesses to  
complete or for a signal from another context or hardware unit.

For example, the programming engines 16 execute eight



contexts by maintaining eight program counters and eight context relative sets of registers. A number of different types of context relative registers, such as general purpose registers (GPRs), inter-programming agent registers, Static Random Access Memory (SRAM) input transfer registers, Dynamic Random Access Memory (DRAM) input transfer registers, SRAM output transfer registers, DRAM output transfer registers. Local memory registers can also be used.

For example, GPRs are used for general programming purposes. GPRs are read and written exclusively under program control. The GPRs, when used as a source in an instruction, supply operands to an execution datapath (not shown). When used as a destination in an instruction, the GPRs are written with the result of the execution box datapath. The programming engines 16 also include IO transfer registers as discussed above. The IO transfer registers are used for transferring data to and from the programming engines 16 and locations external to the programming engines 16, e.g., the DRAM memory 14a and the SRAM memory 14b etc.

#### Bus Architecture:

Referring to FIG. 2, the hardware-based multithreaded processor 12 is shown in greater detail. The DRAM memory 14a and

the SRAM memory 14b are connected to the DRAM memory controller 18a and the SRAM memory 18b, respectively. The DRAM controller 18a is coupled to a pull bus arbiter 30a and a push bus arbiter 32a, which are coupled to a programming engines 16a. The SRAM controller 18b is coupled to a pull bus arbiter 30b and a push bus arbiter 32b, which are coupled to a programming engine 16b. Buses 26a-26b and 28a-28b make up the major buses for transferring data between the programming engines 16a-16b and the DRAM memory 14a and the SRAM memory 14b. Any thread from any of the programming engines 16a-16b can access the DRAM controller 18a and the SRAM controller 18a.

In particular, the push buses 26a-26b have multiple sources of memory such as memory controller channels and internal read registers (not shown) which arbitrate via the push arbiters 32a-32b to use the push buses 26a-26b. The destination (e.g., programming engine 16) of any push data transfer recognizes when the data is being "pushed" into it by decoding the Push\_ID, which is driven or sent with the push data. The pull buses 28a-28b also have multiple destinations (e.g., writing data to different memory controller channels or writeable internal registers) that arbitrate to use the pull buses 28a-28b. The pull buses 28a-28b have a Pull\_ID, which is driven or sent, for example, two cycles before the pull data.

1005736-14566  
Data functions are distributed amongst the programming engines 16. Connectivity to the DRAM memory 14a and the SRAM memory 14b is performed via command requests. A command request can be a memory request. For example, a command request can move data from a register located in the programming engine 16a to a shared resource, e.g., the DRAM memory 14a, SRAM memory 14b. The commands or requests are sent out to each of the functional units and the shared resources. Commands such as I/O commands (e.g., SRAM read, SRAM write, DRAM read, DRAM write, load data from a receive memory buffer, move data to a transmit memory buffer) specify either context relative source or destination registers in the programming engines 16.

In general, the data transfers between programming engines and memory resources designate the memory resource for pushing the data to a processing agent via the push bus having a plurality of sources that arbitrate use of the push bus, and designate the memory resource for receiving the data from the processing agent via the pull bus having a plurality of destinations that arbitrate use of the pull bus.

#### Read Process:

Referring to FIG. 3, a data read process 50 is executed during a read phase of the programming engines 16 by the push

buses 26a-26b. As part of the read process 50 the programming engine executes (52) a context. The programming engine 16 issues (54) a read command to the memory controllers 18a-18b, and the memory controllers 18a-18b processes (56) the request for one of the memory resources, i.e., the DRAM memory 14a or the SRAM memory 14b. For read commands, after the read command is issued (54), the programming engines 16 check (58) if the read data is required to continue the program context. If the read data is required to continue the program context or thread, the context is swapped out (60). The programming engine 16 checks (62) to ensure that the memory controllers 18a-18b have finished the request. When the memory controllers have finished the request, the context is swapped back in (64).

If the request is not required to continue the execution of the context, the programming engine 16 checks (68) if the memory controllers 18a-18b have finished the request. If the memory controllers 18a-18b have not finished the request, a loop back occurs and further checks (58) take place. If the memory controllers 18a-18b have finished the request, when the read data has been acquired from the memory resources, the memory controllers 18a-18b push (70) the data into the context relative input transfer register specified by the read command. The memory controller sets a signal in the programming engine 16 that

enables the context that issued the read to become active. The programming engine 16 reads (72) the requested data in the input transfer register and continues (74) the execution of the context.

5

Write Process:

Referring to FIG. 4, a data write process 80 is executed during a write phase of the programming engines 16 by the pull buses 28a-28b. During the write process 80 the programming engine executes (82) a context. The programming engine 16 loads (84) the data into the output transfer register and issues (86) a write command or request to the memory controllers 18a-18b. The output transfer register is set (88) to a read-only state. For write commands from the programming engines 16, after the output transfer register is set (88) to a read-only state, the programming engine 16 checks (90) if the request is required to continue the program context or thread. If yes, the context is swapped out (92).

If the write request is not required to continue the program context or thread, the memory controllers 18a-18b extracts or pulls (94) the data from the output transfer registers and signals (96) to the programming engines 16 to unlock the output transfer registers. The programming engine 16 then checks (98)

if the context was swapped out. If so, the context is swapped back (100) and if not, the programming engine 16 continues (102) the execution of the context. Thus, the signaled context can reuse the output transfer registers. The signal may also be used to enable the context to go active if it swapped out (100) on the write command.

#### Data Push Operation:

Referring to FIG. 5, a data push operation 110 that occurs in the push buses 26a-26b of the computer processing system 10, is shown in different processing cycles, e.g., cycle 0 through cycle 5. Each target, e.g., the DRAM memory 14a or the SRAM memory 14b, sends or drives (112) a Target\_#\_Push\_ID to the push arbiters where the # indicates the number of different contexts such as context #0 through context #7. The Target\_#\_Push\_ID is derived from the read command and a data error bit (e.g., the numbers following the target represent the source address incrementing in the Push\_ID) for information it would like to push to the push arbiters 32a-32b. For Push\_IDs, each letter indicates a push operation to a particular destination. A Push\_ID destination of "none" indicates that the Push\_ID is null.

The target also sends the Target\_#\_Push\_Data to the Push Arbiter.

The Push\_ID and Push\_Data are registered (114) and enqueued (116) into first-in, first-outs (FIFOs) in the push arbiters 32a-32b unless the Target\_#\_Push\_Q\_Full signal is asserted. This signal indicates that the Push\_ID and Push\_Data FIFOs for that specific target are almost full in the push arbiters 32a-32b. In this case, the push arbiters 32a-32b have not registered a Push\_ID or Push\_Data and the target does not change it. The channel changes the Push\_ID and Push\_Data that is taken by the push arbiters 32a-32b to those for the next word transfer or to null if it has no other valid transfer. Due to latency in the Push\_Q\_Full signal, the push arbiters 32a-32b should accommodate the worst case number of in-flight Push\_IDs and Push\_Data per target.

The push arbiters 32a-32b will arbitrate (118) every cycle between all valid Push\_IDs and send intermediate Push\_ID. The arbitration policy can be round robin, a priority scheme or even programmable. Multiple pushes of data from the push arbiters 32a-32b to the destination are not guaranteed to be in consecutive cycles. The push arbiters 32a-32b send (12) intermediate Push\_Data and Push\_ID is forwarded (120) to the destination. It is up to the target to update the destination address of each Push\_ID it issues for each word of data it wishes to push. The Push\_Data is forwarded (122) to the destination.

At the destination, the time from the destination getting the Push\_ID to the destination getting Push\_Data is fixed by one processing cycle.

## 5 Data Pull Operation:

Referring to FIG. 6, a data pull operation 130 that occurs in the pull buses 28a-28b of the computer processing system 10, is shown in different processing cycles (e.g., cycle 0 through cycle 7). Each target, e.g., the DRAM memory 14a or the SRAM memory 14b, sends or drives (132) the full Target\_#\_Pull\_ID (i.e., the numbers following the target represents the source address incrementing in the Pull\_ID) and length (derived from the write command) for information it would like to pull to the target. For Pull\_IDs, each letter indicates a pull operation from a particular source, e.g., the memory resource 14. A Pull\_ID source of "none" indicates that the Pull\_ID is null. The target must have buffer space available for the pull data when it asserts its Pull\_ID.

The Pull\_ID is registered (134) and enqueued (136) into first-in, first-outs (FIFO) in the pull arbiters 30a-30b, unless the Target\_#\_Pull\_Q\_Full signal is asserted. This signal indicates that the Pull\_ID queue for that specific target is almost full in the pull arbiters 30a-30b. In this case, the pull



arbiters 30a-30b have not registered the Pull\_ID and the target does not change it. The target changes a Pull\_ID that is taken by the pull arbiters 30a-30b to that for the next burst transfer or to null if it has no other valid Pull\_ID. Due to latency in the Pull\_Q\_Full signal, the pull arbiters 30a-30b should accommodate the worst case number of in-flight Pull\_IDs per target.

The pull arbiters 30a-30b arbitrate (138) every cycle among the currently valid Pull\_IDs. The arbitration policy can be round robin, a priority scheme or even programmable.

The pull arbiters 30a-30b forwards (140) the selected Pull\_ID to the source. The time from the pull arbiters 30a-30b sending the Pull\_ID to the source providing data is fixed in three processing cycles. The pull arbiters 30a-30b update the "source address" field of the Pull\_ID for each new data item. The Pull\_Data is pulled (142) from the source and sent to the targets.

The pull arbiters 30a-30b also assert (146) a Target\_#\_Take\_Data to the selected target. This signal is asserted for each cycle a valid word of data is sent to the target. However, the assertions are not guaranteed to be on consecutive processing cycles. The pull arbiters 30a-30b only assert at most one Target\_#\_Take\_Data signal at a time.

For transfers between targets and masters with different bus widths, the pull arbiters 30a-30b are required to do the adjusting. For example, the DRAM controller 18b may accept eight bytes of data per processing cycle but the programming engine 16 may only deliver four bytes per cycle. In this case, the pull arbiters 30a-30b can be used to accept four bytes per processing cycle, merge and pack them into eight bytes, and send the data to the DRAM controller 18a.

Other Embodiments:

It is to be understood that while the example above has been described in conjunction with the detailed description thereof, the foregoing description is intended to illustrate and not limit the scope of the invention, which is defined by the scope of the appended claims. Other aspects, advantages, and modifications are within the scope of the following claims.